Sistemas de Coordenadas

GBC204 – Computação Gráfica
Prof. Dr. rer. nat. Daniel Duarte Abdala

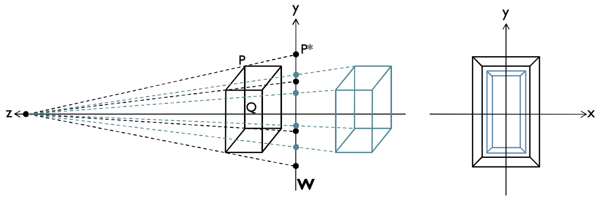
Coordenadas Homogêneas

- Também chamado de coordenadas projetivas;
- Introduzidas por August F. Möbius em 1827;
- Sistema de coordenadas utilizado em geometria projetiva;
- Possui a vantagem de que as coordenadas de pontos, incluindo pontos no infinito, podem ser representados usando coordenadas finitas;
- Fórmulas usando coordenadas homogêneas são frequentemente mais simples e compactas;
- Aplicações:
  - Computação Gráfica
  - Visão Computacional 3D
- Facilita a representação matricial de transformações afins e projetivas.

Coordenadas Homogêneas

- Coordenadas homogêneas requerem uma dimensão a mais que o espaço sendo representado:
  - 1D → 2CH
  - 2D → 3CH
  - 3D → 4CH
- Todas as transformações excetuando-se a translação, resumem-se a multiplicação de uma matriz.
- **DEF**: um ponto $x=<x_1,x_2>/x\in\mathbb{R}^2$ (2D) no sistema de coordenadas cartesianas é representado em coordenadas homogêneas por $[x,y,w]$ onde $x=x_1/w$ e $y = x_2/w$.
- Deste modo, o ponto cartesiano **x** corresponde a uma infinidade de triplas $[x_1/w,\ x_2/w, w]$.
- w é chamado de peso.
- Este sistema permite representação de pontos e direções que tendem ao infinito. Também permite regularizar todas as transformações geométricas vistas até então.

Transformações de Perspectiva



- Próximo ao fim do pipeline gráfico, cenas 3D devem ser convertidas para 2D de modo a ser apresentada em tela;
- A forma como a projeção 3D→2D determina quão realística a cena será;

Transformações de Perspectiva

- A ideia básica refere-se a definir um ponto de observação **E**. A partir de **E** projeta-se uma linha de visão de **E** até cada ponto visível da imagem. O ponto em que a linha de visão intersecciona a janela de visualização **W** indica o ponto de projeção.
- Na figura a linha **EW** está idealmente posicionada perpendicular ao plano de visão.
- O triângulo ΔEWP* é reto e também o é ΔEQP

$$\frac{PQ}{P*W} = \frac{EQ}{EW} \Rightarrow P*W = \frac{PQ \times EW}{EQ}$$

Transformações de Perspectiva

- Tal resultado permite a identificação da posição (coordenadas) de P* na janela de visualização pois se sabe a distância de P* até W.
- Convenções:
  a) A origem coincide com o centro (W) da janela de visualização;
  b) O ponto de vista E é por definição posicionando na parte positiva do eixo-z olhando para a origem. Assim as coordenadas de E são definidas como (0, 0, e);
  c) Os eixos x e y são orientados de modo que o eixo x esteja saindo da figura e o eixo y aponte para cima;

Transformações de Perspectiva

- Assuma que as coordenadas do ponto P sejam (x,y,z);
- Então $EW = e$
  $PQ = y$
  $EQ = e - z$
- Deseja-se identificar o ponto P* na janela de visualização.
- A distância P*W é na realidade a coordenada y do ponto P* na janela de visualização

$$y' = \frac{y \times e}{e - z} = \frac{y}{1 - \frac{z}{e}} \qquad x' = \frac{x}{1 - \frac{z}{e}}$$

Transformações de Perspectiva

- Em suma, a transformação de perspectiva simplesmente divide as coordenadas x e y por (1-z/e);
- Note que se z=0 (2D) a coordenada já está na janela de visualização.

$$Pers(p) = MP = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & -\frac{1}{e} & 1 \end{pmatrix} \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix} = \begin{pmatrix} x \\ y \\ z \\ 1 - \frac{z}{e} = w \end{pmatrix}$$

Translação 2D em Coordenadas Homogêneas

- Assuma p = $\langle p_x, p_y \rangle$ o ponto a ser transladado, e o T = $\langle t_x, t_y \rangle$ vetor de translação.
- Linearmente, representa-se:

$$T(p) = \begin{pmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} p_x \\ p_y \\ w \end{pmatrix} = \begin{pmatrix} p_x + t_x w \\ p_y + t_y w \\ w \end{pmatrix} = p'$$

$$p' = \langle p_x + t_x w, p_y + t_y w \rangle$$

- **p'** possui coordenadas cartesianas: $\langle (p_x / w) + t_x, (p_y / w) + t_y \rangle$

Rotação 2D em Coordenadas Homogêneas

- Assuma p = $\langle p_x, p_y \rangle$ o ponto a ser rotacionado, e θ o ângulo (em readianos) de rotação:
- Linearmente, representa-se:

$$R(p) = \begin{pmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} p_x \\ p_y \\ w \end{pmatrix} = \begin{pmatrix} p_x \cos\theta - p_y \sin\theta \\ p_x \sin\theta + p_y \cos\theta \\ w \end{pmatrix} = p'$$

$$p' = \langle p_x \cos\theta - p_y \sin\theta, p_x \sin\theta + p_y \cos\theta \rangle$$

- Esta operação rotaciona o ponto em relação a origem;
- **p'** possui coordenadas cartesianas: $\langle (p_x / w) + t_x w, (p_y / w) + t_y \rangle$

Encadeamento Rotação↔Translação

- Rotação seguida de translação:

$$RT = \begin{pmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{pmatrix} = \begin{pmatrix} \cos\theta & -\sin\theta & t_x \\ \sin\theta & \cos\theta & t_y \\ 0 & 0 & 1 \end{pmatrix}$$

- Translação seguida de Rotação:

$$TR = \begin{pmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \\ 0 & 0 & 1 \end{pmatrix} = \begin{pmatrix} \cos\theta & -\sin\theta & t_x \cos\theta - t_y \sin\theta \\ \sin\theta & \cos\theta & t_x \sin\theta + t_y \cos\theta \\ 0 & 0 & 1 \end{pmatrix}$$

- A ordem do encadeamento de transformações importa!

Exemplo

- **Desc:** Rotação ao redor de um ponto arbitrário em 2D;
- $\Delta = \{\langle 3,2 \rangle, \langle 4,7 \rangle, \langle 6,1 \rangle\}$, θ=25° ao redor do primeiro ponto <3,2>;
- 1° Transladar o triângulo de modo que <3,2> coincida com a origem **0**=<0,0>;
- 2° Rotacionar ao redor da origem em 25°;
- 3° Transladar de volta para <3,2>;

$$Tu^{-1} R_{25} Tu$$
$$u = \langle t_x, t_y \rangle = \langle -Ap_x, -Ap_y \rangle = \langle -3, -2 \rangle$$

Exemplo

$$\begin{pmatrix}1 & 0 & 3\\ 0 & 1 & 2\\ 0 & 0 & 1\end{pmatrix}\begin{pmatrix}0.91 & -0.42 & 0\\ 0.42 & 0.91 & 0\\ 0 & 0 & 1\end{pmatrix}\begin{pmatrix}1 & 0 & -3\\ 0 & 1 & -2\\ 0 & 0 & 1\end{pmatrix}=\begin{pmatrix}0.91 & -0.42 & 1.11\\ 0.42 & 0.91 & -1.08\\ 0 & 0 & 1\end{pmatrix}$$

$$\langle 3,2\rangle'=\begin{pmatrix}0.91 & -0.42 & 1.11\\ 0.42 & 0.91 & -1.08\\ 0 & 0 & 1\end{pmatrix}\begin{pmatrix}3\\ 2\\ 1\end{pmatrix}=\begin{pmatrix}2.73-0.84+1.11\\ 1.26+1.82-1.08\\ 1\end{pmatrix}=\begin{pmatrix}3\\ 2\\ 1\end{pmatrix}$$

$$\Delta' = \{\langle 3,2\rangle, \langle 1.81, 6.97\rangle, \langle 6.15, 2.35\rangle\}$$

Escalonamento em Coordenadas Homogêneas

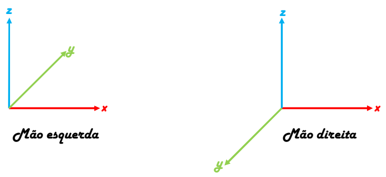
- Assuma p = $\langle p_x, p_y\rangle$ o ponto a ser rotacionado, e k a escala de escalonamento:
- Linearmente, representa-se:

$$S(p)=\begin{pmatrix}k & 0 & 0\\ 0 & k & 0\\ 0 & 0 & 1\end{pmatrix}\begin{pmatrix}p_x\\ p_y\\ w\end{pmatrix}=\begin{pmatrix}kp_x\\ kp_y\\ w\end{pmatrix}=p'$$

$$p'=\langle kp_x, kp_y\rangle$$

- Esta operação escalona o ponto em relação a origem;
- **p'** possui coordenadas cartesianas: $\langle (p'_x/w), (p'_y/w)\rangle$

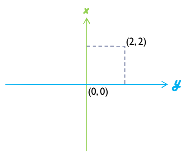
Convenções do Sistema de Coordenadas



- Conversniona-se utilizar sempre o sistema de coordenadas de mão-direita!

Sistema de Coordenadas

- Em OpenGL, quando uma janela é criada para desenho, faz-se necessário que se especifique o sistema de coordenadas desejado e o método de mapeamento das coordenadas do sistema para pixels físicos na tela.
- **Coordenadas Cartesianas 2D**

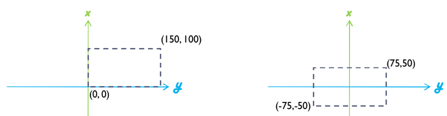


Coordenadas de Clipping

- No domínio do modelo não há necessariamente nenhum limite para quais coordenadas são utilizadas. No entanto, o computador é uma máquina finita e o número de pixels no monitor é igualmente finito.
- Uma janela é medida fisicamente em pixels;
- Antes que qualquer primitiva seja desenhada, faz-se necessário que se especifique qual a parcela do sistema de coordenadas deve ser mapeada para os pixels da janela;
- Está região de mapeamento é chamada de **região de clipping**.

Coordenadas de Clipping

Ex:

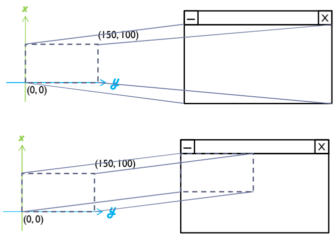


$(x_{min}, y_{min})=(0,0)$
$(x_{max}, y_{max})=(150,100)$

$(x_{min}, y_{min})=(-75,-50)$
$(x_{max}, y_{max})=(75,50)$

Viewports

- Mapeamento de coordenadas do modelo para coordenadas da janela.



Viewports

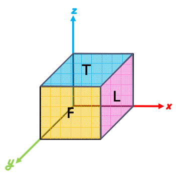
```
def keyboard(key, x, y):
  global wid
  global hei

  if ord(key) == 49:#'1'
    glViewport(0, 0, wid, hei)
    glMatrixMode(GL_PROJECTION)
    glLoadIdentity()
    gluOrtho2D(0.0, wid, 0.0, hei)
    glutPostRedisplay()
  if ord(key) == 50:#'2'
    glViewport(0, 0, wid, hei)
    glMatrixMode(GL_PROJECTION)
    glLoadIdentity()
    gluOrtho2D(-wid/2, wid/2, -hei/2, hei/2)
    glutPostRedisplay()
  if ord(key) == 51: #'3'
    glViewport(0, 0, wid/2, hei/2)
    glMatrixMode(GL_PROJECTION)
    glLoadIdentity()
    gluOrtho2D(0.0, wid, 0.0, hei)
    glutPostRedisplay()
  if ord(key) == 52:#'4'
    glViewport(0, 0, wid/2, hei/2)
    glMatrixMode(GL_PROJECTION)
    glLoadIdentity()
    gluOrtho2D(-wid/2, wid/2, -hei/2, hei/2)
    glutPostRedisplay()
  return
```

- Caso a proporção de tamanhos entre a janela do modelo (window) e a janela do monitor (viewport) seja 1 (aspect ratio = 1) o mapeamento se dá de forma direta;
- Caso AR ≠ 1 a transformação deverá executar uma operação de **interpolação**.

Sistemas de Projeção

- Dois tipos de sistema de projeção Principais:
  - A. Projeção Ortográfica (Ortogonal) (2D)
  - B. Projeção de Perspectiva (3D)

Projeção Ortográfica



- `F – Frente  (x-z)`
- `L – Lateral (y-z)`
- `T – Topo    (x-y)`

```
void gluOrtho2D(left, right, bottom, top)
```

```
void glViewport(x, y, width, height)
```

- gluOrtho2D – define uma matriz de projeção ortográfica 2D;
- glViewport – especifica uma transformação afim de x e y das coordenadas normalizadas do modelo para coordenadas de janela

Projeção Ortográfica

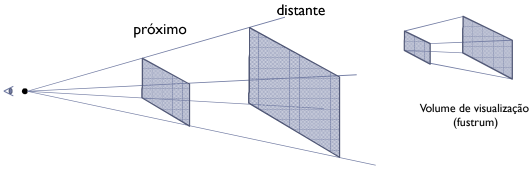
- Seja (x_{nd}, y_{nd}) as coordenadas normalizadas do modelo. As coordenadas de janela (x_w, y_w) são computadas como segue:

$$x_w = (x_{nd} + 1)\left(\frac{width}{2}\right) + x$$

$$y_w = (y_{nd} + 1)\left(\frac{height}{2}\right) + y$$

Projeção de Perspectiva

- A projeção de perspectiva é usada já a muito tempo por artistas e engenheiros para induzir o efeito de tridimensionalidade em desenhors inerentemente bidimensionais;
- A ideia básica é simples. Define-se um ponto de observação. Objetos próximos ao ponto de observação são desenhados com seu tamanho original e objetos mais distantes são desenhados em tamanho menor.




```
void gluPerspective(fovy, aspect, znear, zfar)
```

- **gluPerspective** – especifica um tronco de visualização no sistema de coordenadas do mundo. Em geral, o aspect ratio em gluPerspective deve ser o mesmo que o aspect ratio do viewport associado.

Matriz de Transformação

$$f = \cot\left(\frac{fovy}{2}\right)$$

$$A = \begin{pmatrix} f / aspect & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & {zFar + zNear}/{zNear - zFar} & {2 \times zFar \times zNear}/{zNear - zFar} \\ 0 & 0 & -1 & 0 \end{pmatrix}$$